DUBLIN CITY UNIVERSITY

ELECTRONIC AND COMPUTER ENGINEERING

Streaming Audio Server with Listener-Tracking Embedded Clients

Authors

Michael Lenehan   michael.lenehan4@mail.dcu.ie

Supervisor

Martin Collier   martin.collier@mail.dcu.ie

08/04/2019

Table of Contents

1 Acknowledgmenets

I would like to thank my project supervisor, Dr. Martin Collier. He provided guidance and assistance with all aspects of this project, and his knowledge and insight on the topic were invaluable. I would also like to thank Dr. Gabriel-Miro Muntean, who acted as the second assessor for this project. His insight into testing practices, which gave greater clarity to the overall project results.

2 Declaration

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

I have read and understood the DCU Academic Integrity and Plagiarism at https://www4.dcu.ie/sites/default/files/policy/1%20-%20integrity_and_plagiarism_ovpaa_v3.pdf and IEEE referencing guidelines found at https://loop.dcu.ie/mod/url/view.php?id=448779.

Name: Michael Lenehan Date: 08/04/2019

3 Abstract

Open-Source audio server softwares are numerous. The most popular available options for embedded linux devices are intended for use as "headless" audio devices, controlled over the network, and outputting their audio locally. With a large number of subscription based audio streaming services available, such as Spotify, Tidal, and Google Play Music, there is an increasing need for an option to stream a users own music, from a central storage device, to their client device of choice.

The addition of listener tracking allows a user to play audio through multiple speakers as they traverse a space, such as their home, without needing to play music through all available speakers, or without turning device volumes up or down.

4 Implementation and Testing

4.1 Server Software Testing

A number of parameters must be tested in order to determine the optimal open source audio server solution. Each audio server software is tested under the same testing conditions, and the values for network usage, CPU temperature, CPU load, and CPU frequency are monitored and recorded.

The testing setup consists of three Raspberry Pi's, each running the Raspbian Stretch Light OS. One Raspberry Pi runs the audio server software, and the snapcast server software. The second Raspberry Pi runs the snapclient software. The final Raspberry Pi runs the Munin server software, allowing to monitor the clients, which are running on the other two Raspberry Pi's.

Munin

Munin is a server performance monitoring software, which runs on an Apache server, with the client software running on each device requiring monitoring. The recorded information is hosted on a locally accessible website, at the IPp address of the server device. The output information is displayed in graphical representation, which can be analyzed.

Cron

Cron is a scheduling utility, which allows for the automation of command execution at specified times, or set time intervals. Using a crontable, a file for entering cron jobs, the required testing schedule can be run on the audio server Raspberry Pi. For the purposes of testing the audio server software while streaming audio files of different formats, a crontable is configured to play audio in the .wav format, followed by audio in the .flac format, followed by audio in the .mp3 format. Each audio format is played continuously for two hours, with a two minute space between formats.

4.2 Audio Server Testing Procedure

The testing steps outlined below must be repeated for each of the available audio formats.

1. Create a crontable on the server device to start and stop audio playback at a set time(s):

   - 00 10 * * * mpc add *.wav && mpc repeat on && mpc play

   - 00 12 * * * mpc stop && mpc clear && mpc repeat off

   - 02 12 * * * mpc add *.flac && mpc repeat on && mpc play

   - 02 14 * * * mpc stop && mpc clear && mpc repeat off

   - 04 14 * * * mpc add *.mp3 && mpc repeat on && mpc play

   - 04 16 * * * mpc stop && mpc clear && mpc repeat off

2. Once testing is complete of all available audio formats, replace the audio server software - "mpd" - with the "mopidy" audio server software. Repeat step one for the "mopidy" server software.

3. Once testing is complete of all available audio formats, replace the audio server software - "mopidy" - with the "Volumio" audio server software. Repeat step one for the "Volumio" server software.

4. Record the data output from Munin.

5 Results and Analysis

5.1 Results

5.2 Analysis

6 Ethics

6.1 Audio Tracks

While there have been questions raised over the ethical problems with streaming services, and the revenue paid to artists whose music is played via their platforms, there is also a number of issues in stored audio, with users illegally downloading audio tracks.

6.2 Listener Tracking

There are a number of ethical issues involved with client tracking in services such as that described in this project.